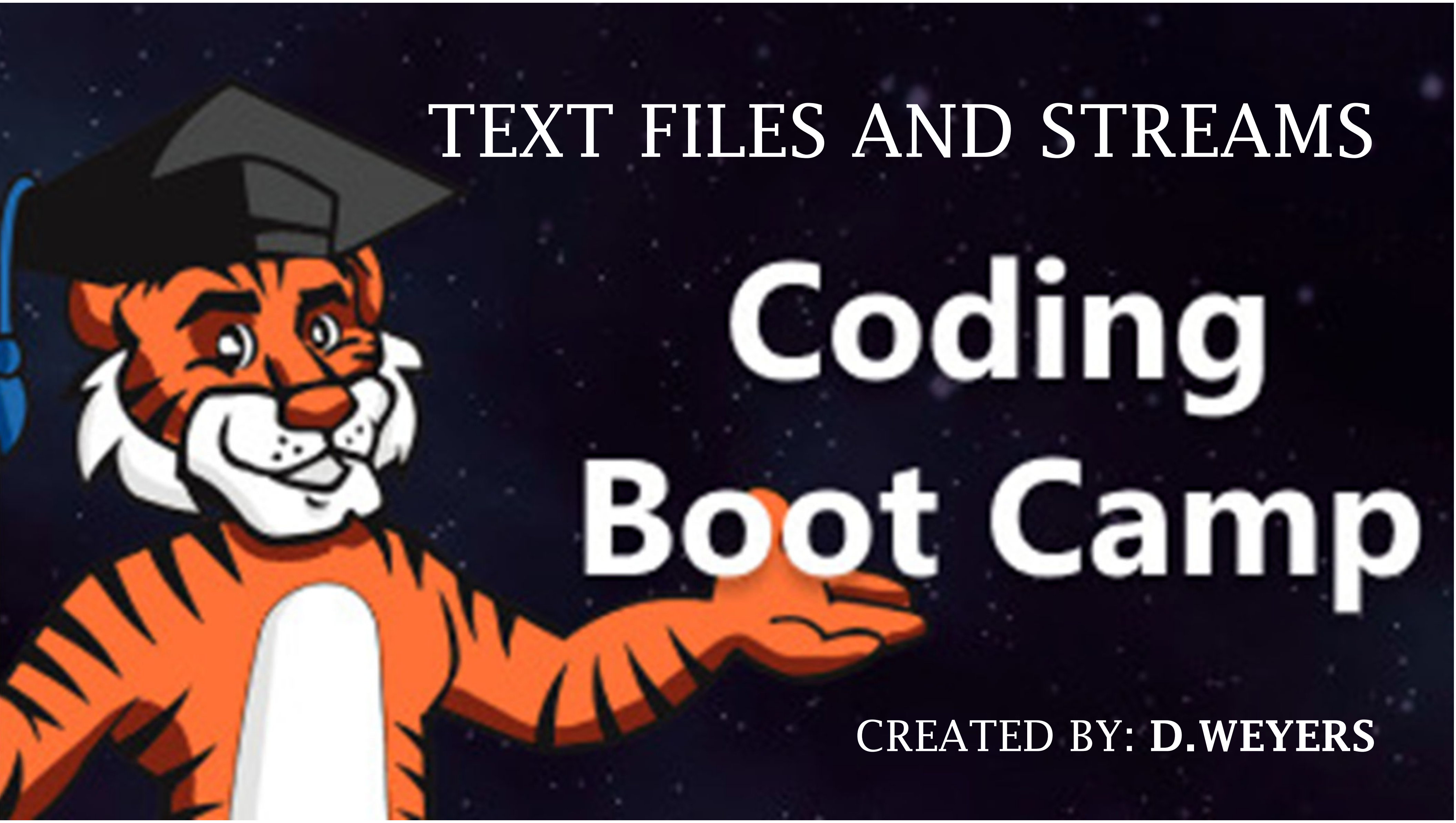


TEXT FILES AND STREAMS

CREATED BY: **D.WEYERS**

Text file

WHAT IS A TEXT FILE?

A text file contains text with no formatting, that is, no formatting features such as bold, underline, tables, styles and so on. Since the text file has no formatting, it can be used by different programs.

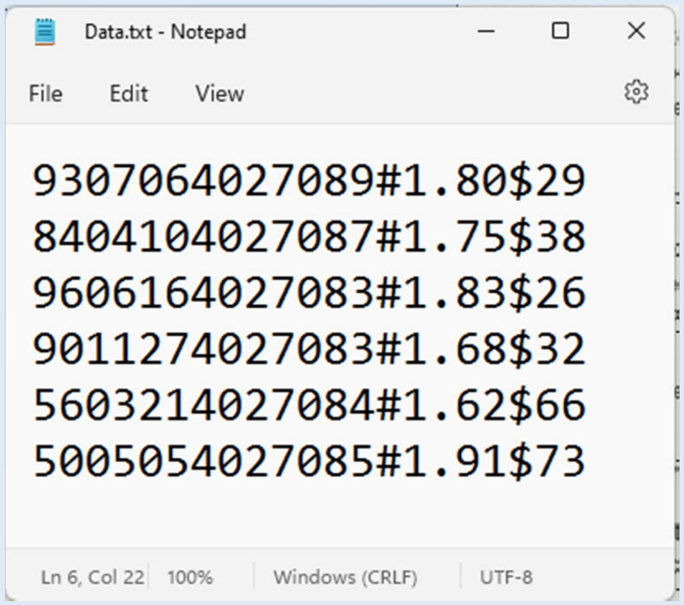
CREATING A TEXT FILE

You can create a text file by using any one of the methods below:

- A text editor like NotePad:
  Open the NotePad program, type the text and save the file. The file will have a .txt extension.
- The Delphi Code Editor:
  In Delphi:
  - Click on *File*, *New*, *Other*, *Other file*, *Text File*, *OK*. Select *.txt as the file extension.
  - Type the text and save the file.
- Writing Delphi Code.

Note:

- The text file must be saved in the same folder as the project file.

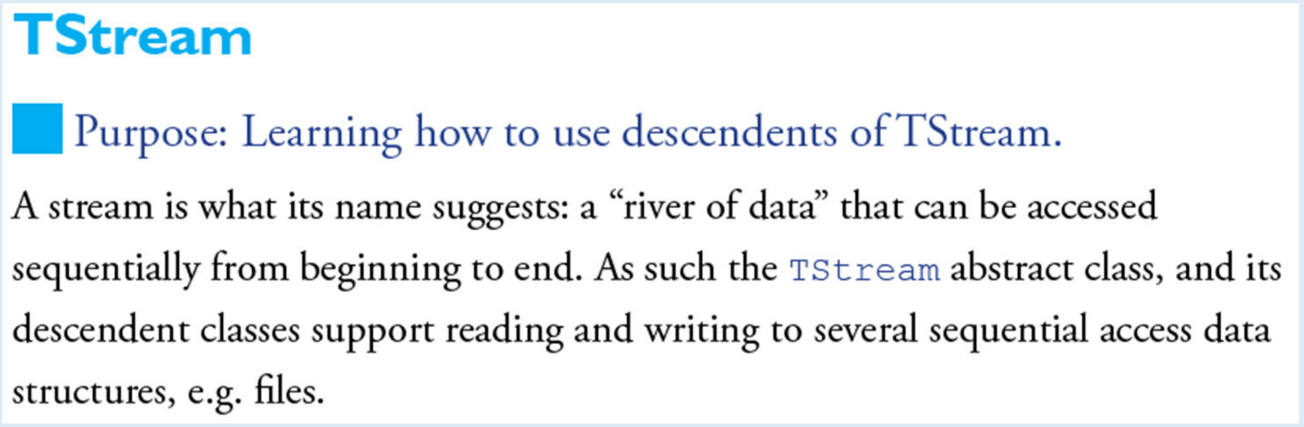


```
9307064027089#1.80$29
8404104027087#1.75$38
9606164027083#1.83$26
9011274027083#1.68$32
5603214027084#1.62$66
5005054027085#1.91$73
```

Procedures used:

AssignFile	– assigns file name to file variable
Reset	– Set focus to start of file; file ready to be read
Readln	– Reads line of file and stores it in a string variable
CloseFile	– closes file for reading or writing
Append	– opens up already created file and places focus at the end of the file to add new text
Rewrite	– creates new file and sets focus to start of file

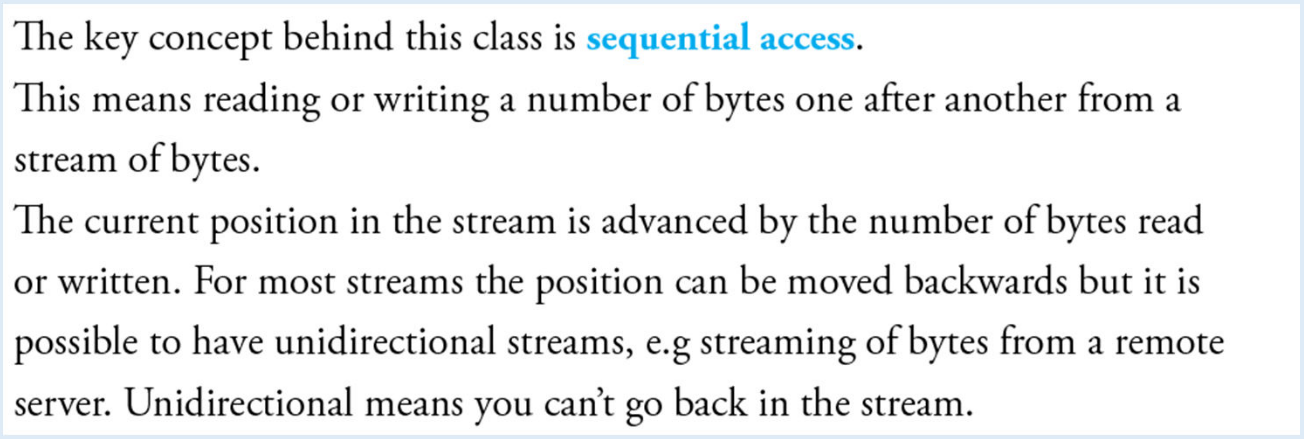
Streams



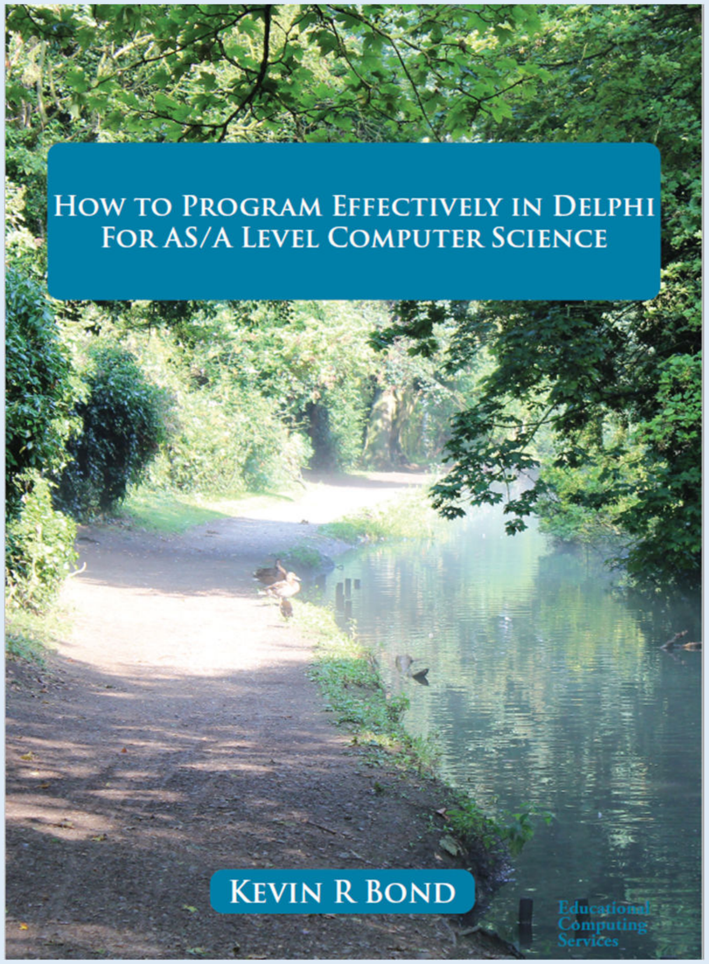
TStream

Purpose: Learning how to use descendents of TStream.

A stream is what its name suggests: a "river of data" that can be accessed sequentially from beginning to end. As such the `TStream` abstract class, and its descendent classes support reading and writing to several sequential access data structures, e.g. files.



The key concept behind this class is **sequential access**.
This means reading or writing a number of bytes one after another from a stream of bytes.
The current position in the stream is advanced by the number of bytes read or written. For most streams the position can be moved backwards but it is possible to have unidirectional streams, e.g streaming of bytes from a remote server. Unidirectional means you can't go back in the stream.


How to Program Effectively in Delphi
For AS/A Level Computer Science
Kevin R Bond
Educational Computing Services

***How to Program Effectively in Delphi For AS/A Level Computer Science**
Kevin R Bond

READING FROM A TEXT FILE *ALGORITHM*

Step 1: Declare

```
var
  MyRfile : textfile;
  sOneline : string;
  iPlace, iCounter : integer;
```

Step 2: Test

```
if FileExists('Data.txt') = false then
  begin
    ShowMessage('File not Found');
    Exit;
  end;
```

Returns a Boolean value that indicates whether a file with the given path exists or not.

Step 3: Assign

```
AssignFile(MyRfile,'Data.txt');
```

Creates a link between the logical file name and the physical file found on a storage medium.

Step 4: Reset

```
Reset(MyRfile);
```

Opens an existing file for read only access and sets the file pointer at the beginning of the file

Text file:

```
9307064027089#1.80$29
8404104027087#1.75$38
9606164027083#1.83$26
```

```
iCounter := 0;
```

Step 5: Loop

READING FROM A TEXT FILE * ALGORITHM *

Step 6: Read

```
Readln(MyRfile,sOneline);
```

Reads the selected line of text from the current file pointer position in the file and moves the file pointer to

```
9307064027089#1.80$29
```

```
ID : string;
Height : Real;
Age : integer;
```

Step 7: Extract

```
7.1  iPlace := Pos('#',sOneline);
7.2  sxID := Copy(sOneline,1, iPlace-1);
7.3  Delete(sOneline,1, iPlace);
```

```
1.80$29
```

```
7.1  iPlace := Pos('$',sOneline);
7.2  rxHeight := StrToFloat(Copy(sOneline,1, iPlace-1));
7.3  Delete(sOneline,1, iPlace);

     ixAge := StrToInt(sOneline);
```

```
29
```

Step 8: Process\Display

```
redDisplay.Lines.Add(sxID + #9 + FloatToStr(rxHeight) + #9 + inttostr(ixAge));
```

Step 9: Close

```
Closefile(MyRfile);
```

Closes the link between the text file's logical name and the physical file name. You cannot read or write to the text file once it is closed.

READING FROM A TEXT FILE * ALGORITHM *

```
private
  arrxID : array[1..30] of string;
  arrxHeight : array[1..30] of real;
  arrxAge : array[1..30] of integer;
  iComplete : integer;
```

Step 6: Read

```
Readln(MyRfile,sOneline);
```

Step 7: Extract

```
inc(iCounter);
```

```
7.1  iPlace  := Pos('#',sOneline);
7.2  arrxID[iCounter]  := Copy(sOneline,1, iPlace-1);
7.3  Delete(sOneline,1, iPlace);

7.1  iPlace  := Pos('$',sOneline);
7.2  arrxHeight[iCounter]  := StrToFloat(Copy(sOneline,1, iPlace-1));
7.3  Delete(sOneline,1, iPlace);

     arrxAge[iCounter]  := StrToInt(sOneline);
```

Step 9: Close

```
Closefile(MyRfile);
iComplete  := iCounter;
```

Display out

```
for K := 1 to iComplete do
  redDisplay.Lines.Add(arrxID[K] + #9 + FloatToStr(arrxHeight[K]) + #9 + inttostr(arrxAge[K]));
```

WRITING TO A TEXT FILE * ALGORITHM *

Step 1: Declare

```
var
  MyWfile : textfile;
  sOneline : string;
```

Step 2: Assign

```
Assignfile(MyWfile,'Save.txt');
```

Step 3: Test

```
if FileExists('Save.txt') =
  rewrite(MyWfile)
else
  append(MyWfile);
```

Create a new file w

Opens the file for wri
the file pointer to the
so that text can be w
end of the text file

Step 4: Write

```
sOneline := '';

Writeln(MyWfile,sOneline);
```

Writes a line
position of the
end-of-line m
text.

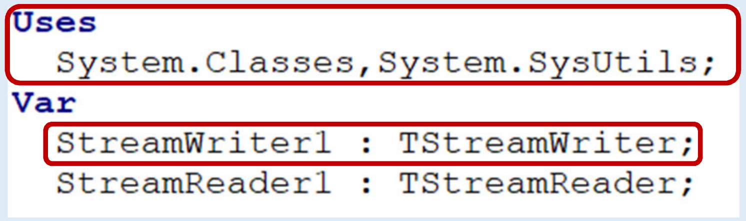
Step 5: Close

```
Closefile(MyWfile);
```

```
var
  MyWfile : textfile;
  sOneline : string;
begin

  Assignfile(MyWfile,'Save.txt');

  if FileExists('Save.txt') = false then
    rewrite(MyWfile)
  else
    append(MyWfile);

  sOneline := '';
  Writeln(MyWfile,sOneline);

  Closefile(MyWfile);
```

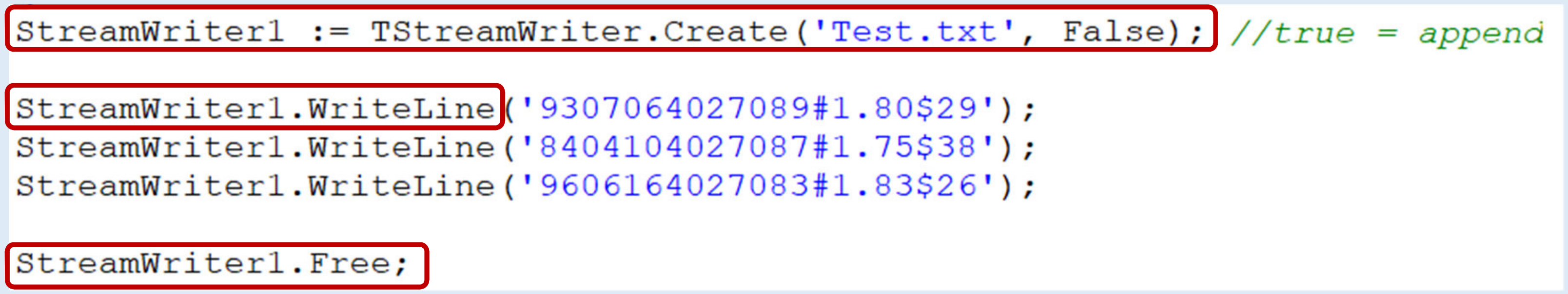
STREAMS WRITE * ALGORITHM *

Step 1: Declare



```
Uses
  System.Classes,System.SysUtils;
Var
  StreamWriter1  : TStreamWriter;
  StreamReader1  : TStreamReader;
```

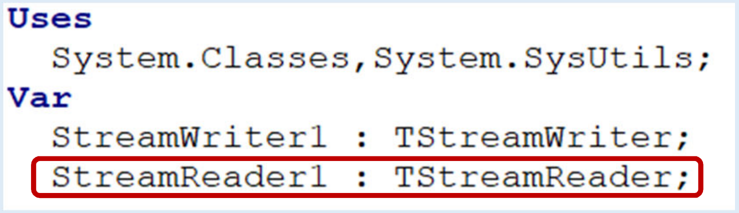
Step 2: Write



```
StreamWriter1 := TStreamWriter.Create('Test.txt', False); //true = append

StreamWriter1.WriteLine('9307064027089#1.80$29');
StreamWriter1.WriteLine('8404104027087#1.75$38');
StreamWriter1.WriteLine('9606164027083#1.83$26');

StreamWriter1.Free;
```

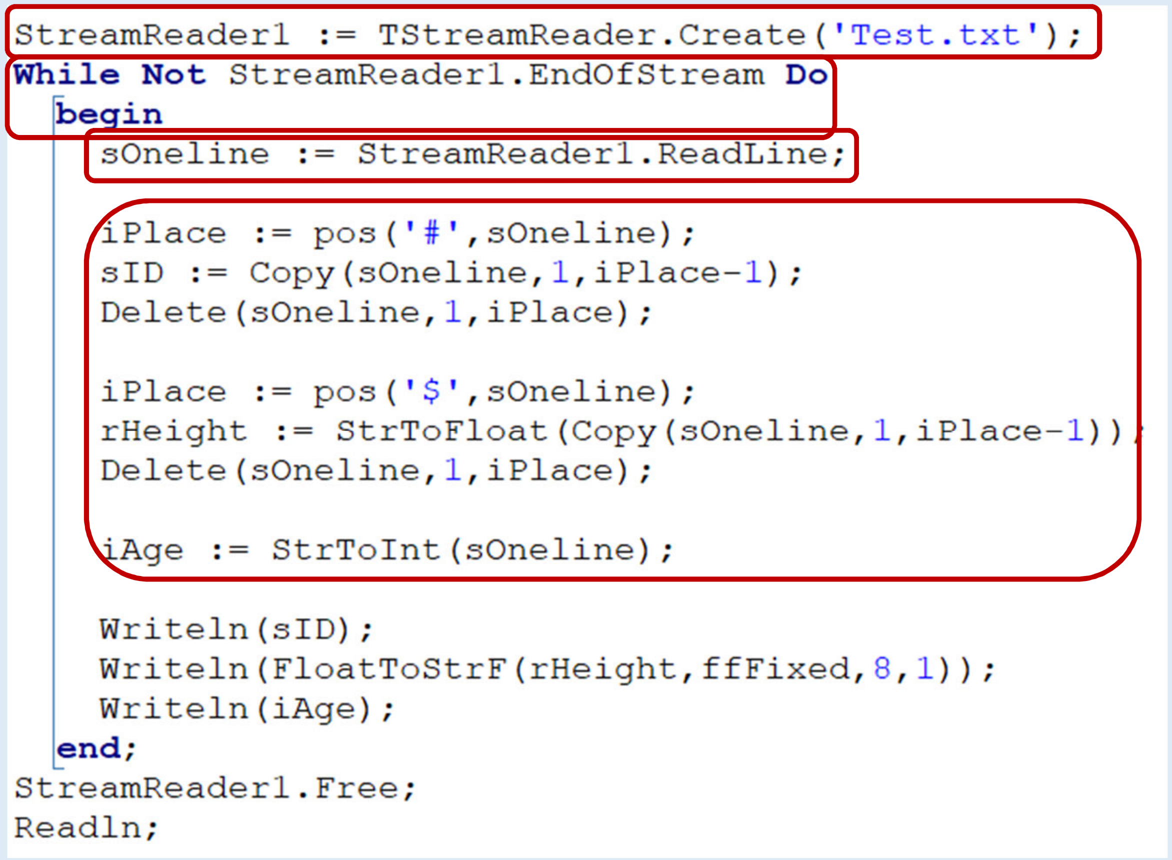
STREAMS READ * ALGORITHM *

Step 1: Declare



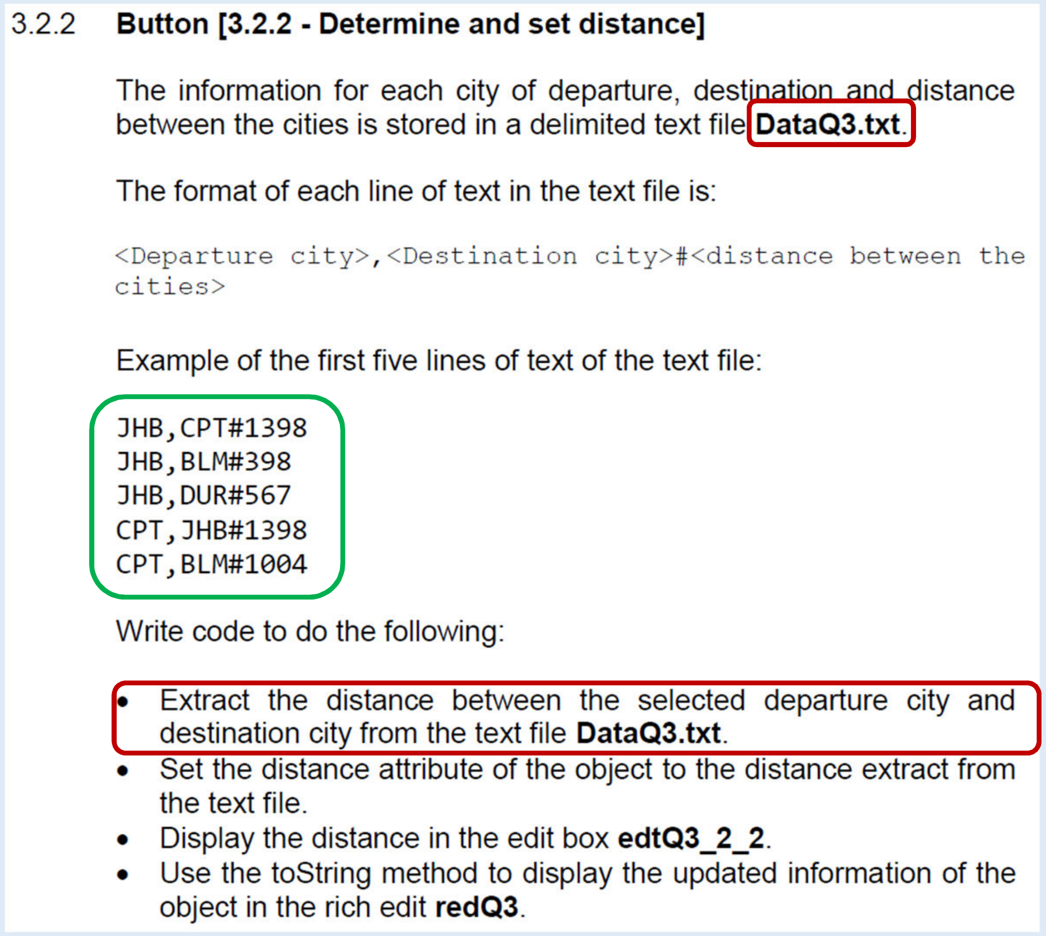
```
Uses
  System.Classes,System.SysUtils;
Var
  StreamWriter1 : TStreamWriter;
  StreamReader1 : TStreamReader;
```

Step 2: Read



```
StreamReader1 := TStreamReader.Create('Test.txt');
While Not StreamReader1.EndOfStream Do
  begin
    sOneline := StreamReader1.ReadLine;

    iPlace := pos('#',sOneline);
    sID := Copy(sOneline,1,iPlace-1);
    Delete(sOneline,1,iPlace);

    iPlace := pos('$',sOneline);
    rHeight := StrToFloat(Copy(sOneline,1,iPlace-1));
    Delete(sOneline,1,iPlace);

    iAge := StrToInt(sOneline);

    Writeln(sID);
    Writeln(FloatToStrF(rHeight,ffFixed,8,1));
    Writeln(iAge);
  end;
StreamReader1.Free;
Readln;
```

EXAMPLE'S OF TEXT FILE QUESTIONS



3.2.2 **Button [3.2.2 - Determine and set distance]**

The information for each city of departure, destination and distance between the cities is stored in a delimited text file **DataQ3.txt**.

The format of each line of text in the text file is:

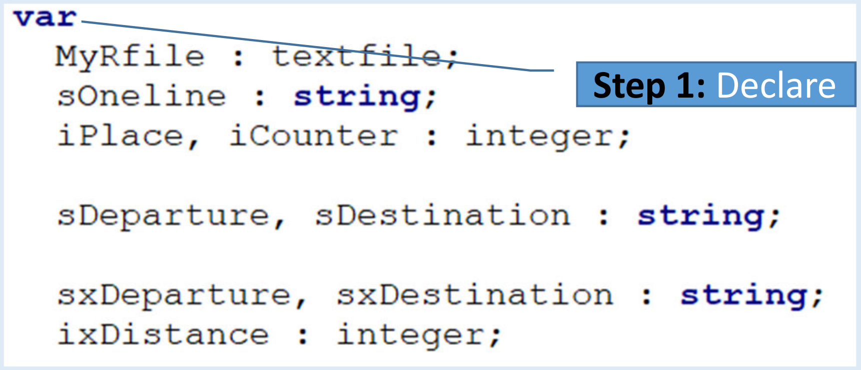
```
<Departure city>,<Destination city>#<distance between the
cities>
```

Example of the first five lines of text of the text file:

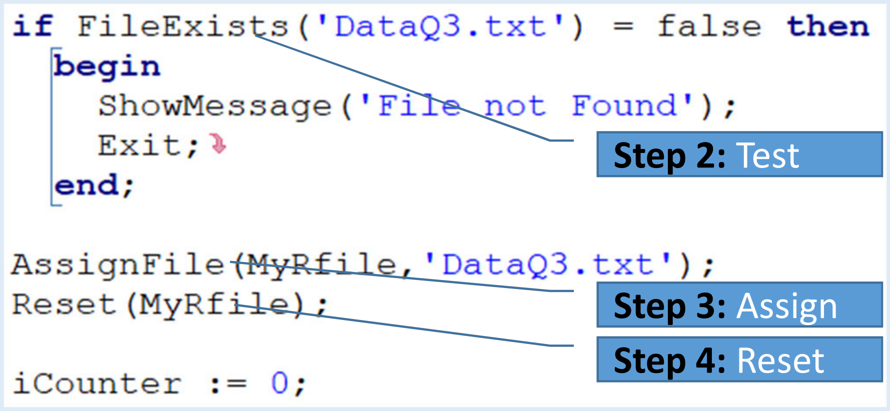
```
JHB,CPT#1398
JHB,BLM#398
JHB,DUR#567
CPT,JHB#1398
CPT,BLM#1004
```

Write code to do the following:

- Extract the distance between the selected departure city and destination city from the text file **DataQ3.txt**.
- Set the distance attribute of the object to the distance extract from the text file.
- Display the distance in the edit box **edtQ3_2_2**.
- Use the toString method to display the updated information of the object in the rich edit **redQ3**.

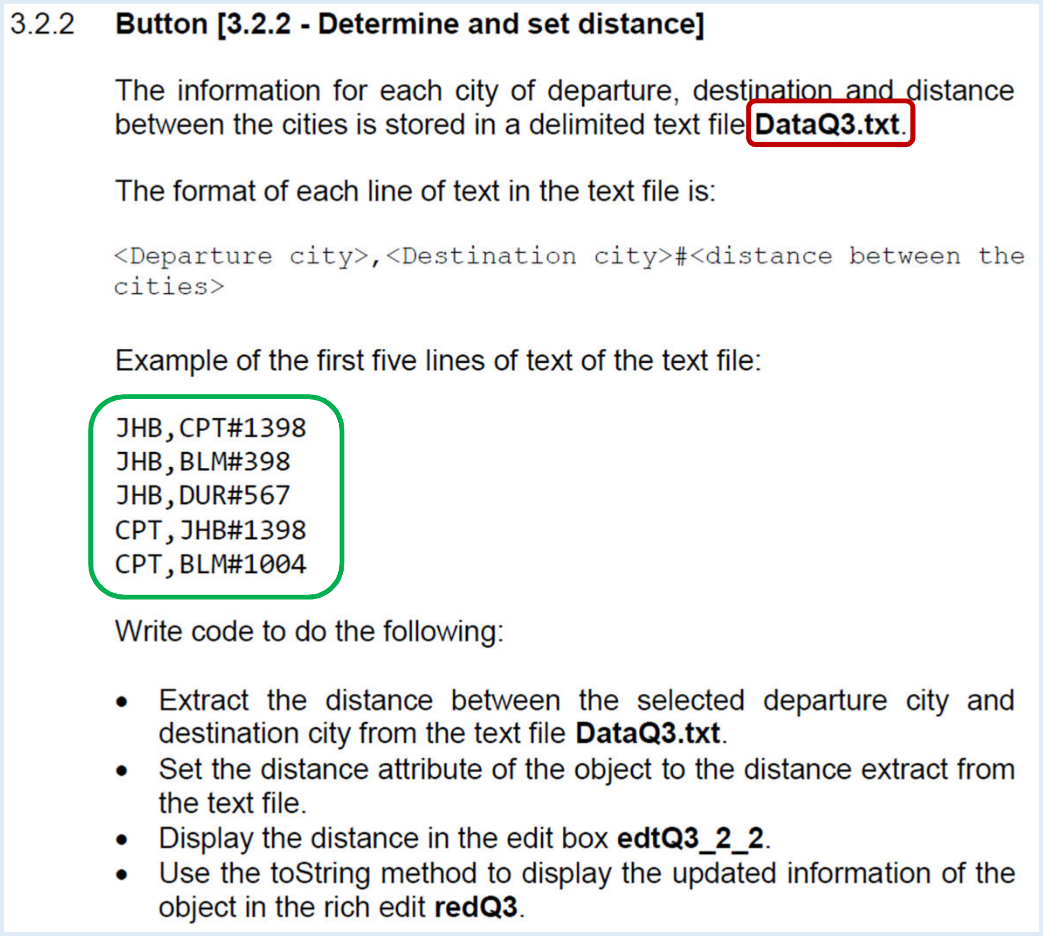


```
var
  MyRfile : textfile;
  sOneline : string;
  iPlace, iCounter : integer;

  sDeparture, sDestination : string;

  sxDeparture, sxDestination : string;
  ixDistance : integer;
```



```
if FileExists('DataQ3.txt') = false then
  begin
    ShowMessage('File not Found');
    Exit;
  end;

AssignFile(MyRfile,'DataQ3.txt');
Reset(MyRfile);

iCounter := 0;
```

EXAMPLE'S OF TEXT FILE QUESTIONS



3.2.2 **Button [3.2.2 - Determine and set distance]**

The information for each city of departure, destination and distance between the cities is stored in a delimited text file **DataQ3.txt**.

The format of each line of text in the text file is:

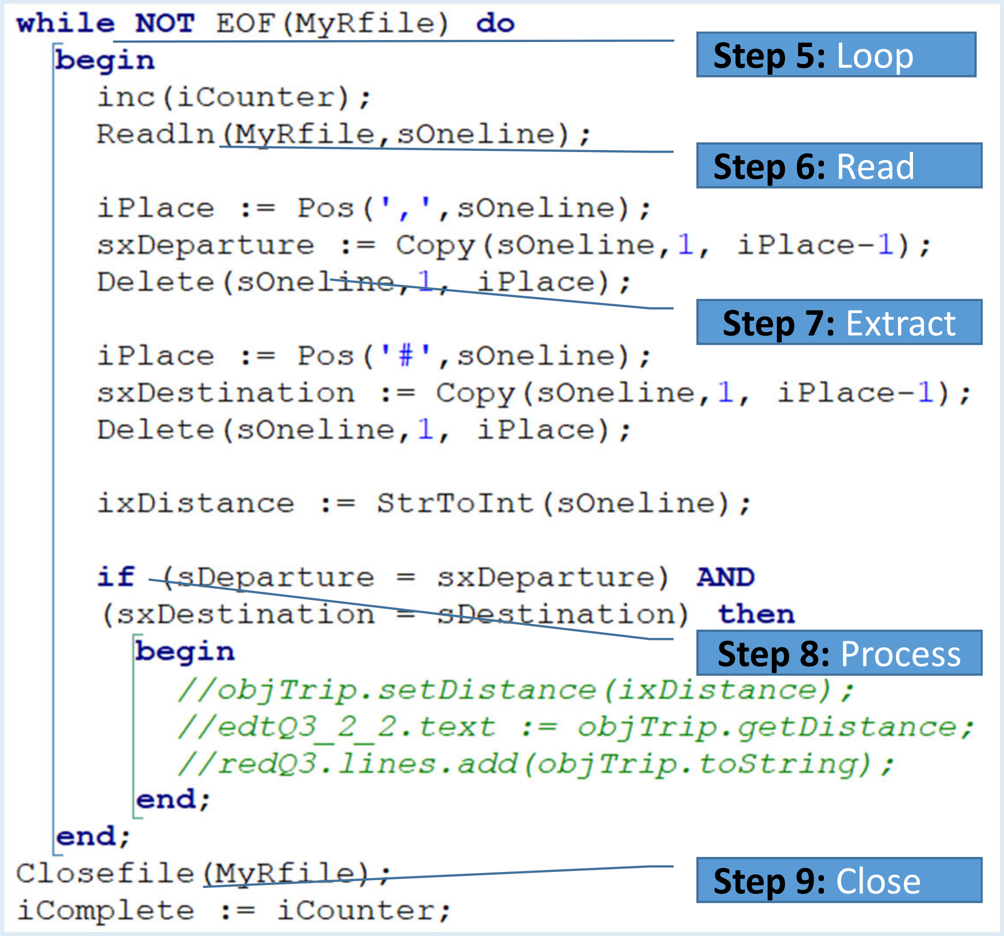
```
<Departure city>,<Destination city>#<distance between the
cities>
```

Example of the first five lines of text of the text file:

```
JHB,CPT#1398
JHB,BLM#398
JHB,DUR#567
CPT,JHB#1398
CPT,BLM#1004
```

Write code to do the following:

- Extract the distance between the selected departure city and destination city from the text file **DataQ3.txt**.
- Set the distance attribute of the object to the distance extract from the text file.
- Display the distance in the edit box **edtQ3_2_2**.
- Use the toString method to display the updated information of the object in the rich edit **redQ3**.



```
while NOT EOF(MyRfile) do
 begin
   inc(iCounter);
   Readln(MyRfile,sOneline);

   iPlace := Pos(',',sOneline);
   sxDeparture := Copy(sOneline,1, iPlace-1);
   Delete(sOneline,1, iPlace);

   iPlace := Pos('#',sOneline);
   sxDestination := Copy(sOneline,1, iPlace-1);
   Delete(sOneline,1, iPlace);

   ixDistance := StrToInt(sOneline);

   if (sDeparture = sxDeparture) AND
   (sxDestination = sDestination) then
     begin
       //objTrip.setDistance(ixDistance);
       //edtQ3_2_2.text := objTrip.getDistance;
       //redQ3.lines.add(objTrip.toString);
     end;
 end;
Closefile(MyRfile);
iComplete := iCounter;
```

- The folder also contains a **text file** named: **Data.txt.** It contains the name and three marks of several learners. The information for each learner is provided in the file in the following format:
  Name Surname#Marks1#Mark2#Mark3

3.2.1 **Button [3.2.1]**

The user must enter the name of the learner. The program must find the name of the learner in the text file named Data.txt.

Write code to do the following:

- Check whether the text file Data.txt exists or not. If the text file does NOT exist, display a suitable message and close the application.
- If the text file exists, search for the learner's name in the text file.
- If the learner's name is found in the text file:
  - Extract the learner's three marks
  - Use a Boolean variable to stop the search process immediately.
  - Instantiate the objMarks object with the required values.

```
var
  sName : string;
  Myler : TextFile;
  sOneline : string;
  sTname : string;
  iMark1,iMark2,iMark3 : Integer;
  iPlace : Integer;
  bFlag : Boolean;
```

```
begin
 sName := edt.Text;

 if FileExists('Data.txt') <> true then
   begin
     MessageDlg('Text file does NOT exist',mtError,mbYesNo,0);
     Exit;
   end;
```

```
AssignFile(Myler,'Data.txt');
reset(Myler);

bFlag := False;
```

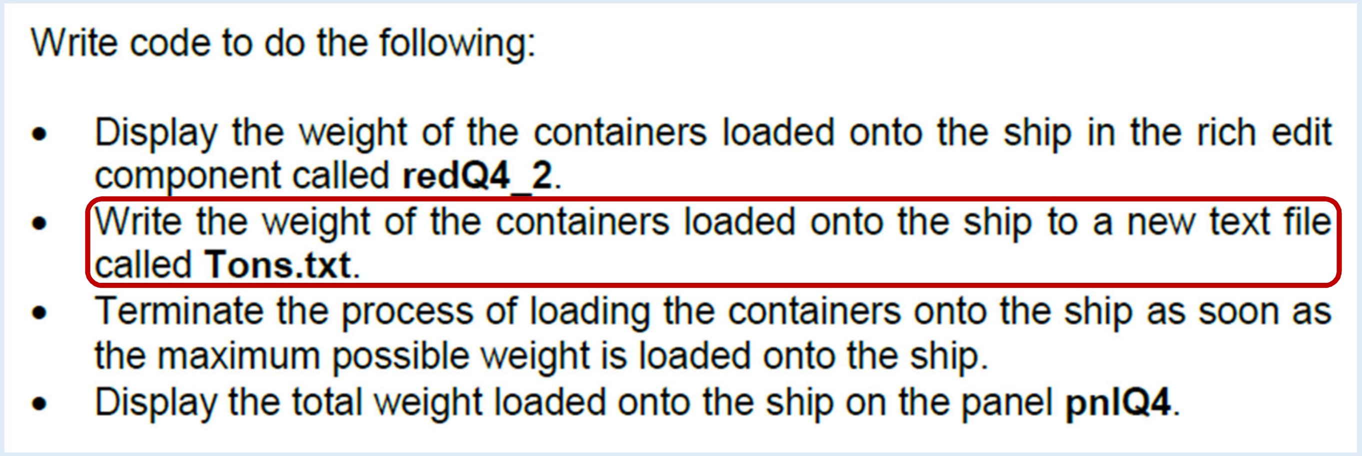
```
while NOT(EOF(myler)) AND (bFlag = False) do
  begin
    readln(myler,sOneline);   //Gert Combrink#77#82#81

    iPlace := Pos('#',sOneline);
    sTname := Copy(sOneline,1,iPlace-1);
    Delete(sOneline,1,iPlace);
```

```
 if sTname = sName then
   begin
     bFlag := True;
```

```
iPlace := Pos('#',sOneline);
iMark1 := StrToInt(Copy(sOneline,1,iPlace-1));
Delete(sOneline,1,iPlace);

iPlace := Pos('#',sOneline);
iMark2 := StrToInt(Copy(sOneline,1,iPlace-1));
Delete(sOneline,1,iPlace);

iMark3 := StrToInt(sOneline);
```

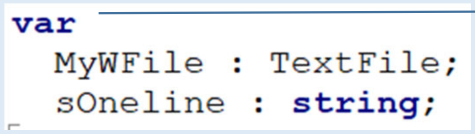
```
objMarks := TMarks.Create(sTname,iMark1,iMark2,iMark3);
```

EXAMPLE'S OF TEXT FILE QUESTIONS



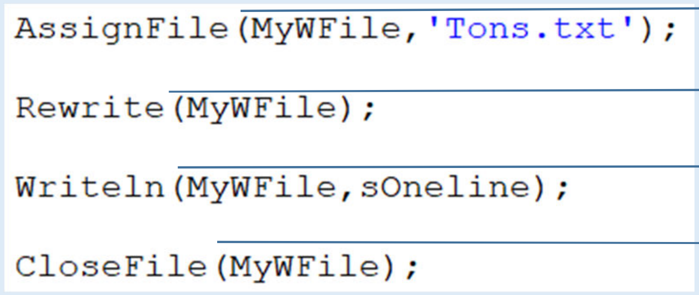
Write code to do the following:

- Display the weight of the containers loaded onto the ship in the rich edit component called **redQ4_2**.
- Write the weight of the containers loaded onto the ship to a new text file called **Tons.txt**.
- Terminate the process of loading the containers onto the ship as soon as the maximum possible weight is loaded onto the ship.
- Display the total weight loaded onto the ship on the panel **pnlQ4**.



```
var
  MyWFile : TextFile;
  sOneline : string;
```

Step 1: Declare



```
AssignFile(MyWFile,'Tons.txt');
```

Step 2: Assign

```
Rewrite(MyWFile);
```

Step 3: Test

```
Writeln(MyWFile,sOneline);
```

Step 4: Write

```
CloseFile(MyWFile);
```

Step 5: Close

THANK YOU
Coding
Boot Camp
CREATED BY: D.WEYERS